OverBlownINS: The Incompressible Navier–Stokes Solver in **OverBlown**

William D. Henshaw
Centre for Applied Scientific Computing
Lawrence Livermore National Laboratory
Livermore, CA, 94551.
henshaw@llnl.gov
http://www.llnl.gov/casc/people/henshaw
http://www.llnl.gov/casc/Overture

November 6, 2003

Abstract:

This document describes **OverBlownINS**, a solver written using the **Overture** framework to solve the incompressible Navier-Stokes (INS). **OverBlownINS** is part of the **OverBlown** solver. The INS solver can be used to the solve time-dependent Navier-Stokes equations to second and fourth-order accuracy. There is also a pseudo-steady line implicit solver with a nonlinear second- or fourth-order artificial dissipation.

Contents

1 Introduction

This document is currently under development.

OverBlownINS is a solver for the incompressible Navier-Stokes equations. **OverBlown** is a fluid flow solver for overlapping grids built upon the **Overture** framework [1],[4],[2].

2 The Equations

The incompressible Navier-Stokes equations are

$$\mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p = \nu\Delta\mathbf{u}, \tag{1}$$
$$\nabla\cdot\mathbf{u} = 0. \tag{2}$$

We solve the incompressible Navier-Stokes equations written in the form (pressure-poisson system)

$$\left.\begin{aligned} \mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p - \nu\Delta\mathbf{u} - \mathbf{f} &= 0 \\ \Delta p - (\nabla u\cdot\mathbf{u}_x + \nabla v\cdot\mathbf{u}_y + \nabla w\cdot\mathbf{u}_z) - C_d(\nu)\nabla\cdot\mathbf{u} - \nabla\cdot\mathbf{f} &= 0 \end{aligned}\right\} \quad \mathbf{x}\in\Omega \tag{3}$$
$$\left.\begin{aligned} B(\mathbf{u},p) &= 0 \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right\} \quad \mathbf{x}\in\partial\Omega$$
$$\mathbf{u}(\mathbf{x},0) = \mathbf{u}_0(\mathbf{x}) \qquad \text{at } t=0$$

There are n_d boundary conditions, $B(\mathbf{u},p)=0$, where n_d is the number of space dimensions. On a no-slip wall, for example, $\mathbf{u}=0$. In addition, a boundary condition is required for the pressure. The boundary condition $\nabla\cdot\mathbf{u}=0$ is added. With this extra boundary condition it follows that the above problem is equivalent to the formulation with the Poisson equation for the pressure replaced by $\nabla\cdot\mathbf{u}=0$ everywhere. The term $C_d(\nu)\nabla\cdot\mathbf{u}$ appearing in the equation for the pressure is used to damp the divergence [5]. For further details see also [3]

3 Discretization

Let $\mathbf{V}_i$ and P_i denote the discrete approximations to $\mathbf{u}$ and p so that

$$\mathbf{V}_i \approx \mathbf{u}(\mathbf{x}_i) \quad , \quad P_i \approx p(\mathbf{x}_i) \ .$$

Here $\mathbf{V}_i = (V_{1i}, V_{2i}, V_{3i})$ and $i=(i_1,i_2,i_3)$ is a multi-index. After discretizing in space the equations we solve are of the form

$$\left.\begin{aligned} \tfrac{d}{dt}\mathbf{V}_i + (\mathbf{V}_i\cdot\nabla_h)\mathbf{V}_i + \nabla_h P_i - \nu\Delta_h\mathbf{V}_i - \mathbf{f}(\mathbf{x}_i,t) &= 0 \\ \Delta_h P_i - \textstyle\sum_m \nabla_h V_{m,i}\cdot D_{m,h}\mathbf{V}_i - C_{d,i}\nabla_h\cdot\mathbf{V}_i - \nabla_h\cdot\mathbf{f}(\mathbf{x}_i,t) &= 0 \end{aligned}\right\} \quad \mathbf{x}\in\Omega$$
$$\left.\begin{aligned} B(\mathbf{V}_i,P_i) &= 0 \\ \nabla_h\cdot\mathbf{V}_i &= 0 \end{aligned}\right\} \quad \mathbf{x}_i\in\partial\Omega_h$$
$$\mathbf{V}(\mathbf{x}_i,0) = \mathbf{U}_0(\mathbf{x}_i) \qquad \text{at } t=0$$

where the divergence damping coefficient, $C_{d,i}$ is defined below. The subscript “h” denotes a second or fourth-order centred difference approximation,

$$D_{m,h} \approx \frac{\partial}{\partial x_m} \quad , \quad \nabla_h = (D_{1,h}, D_{2,h}, D_{3,h}) \quad , \quad \Delta_h \approx \sum_m \frac{\partial^2}{\partial x_m^2}$$

Extra numerical boundary conditions are also added, see [3] [8] for further details. An artificial diffusion term can be added to the momentum equations. This is described in section (5).

When discretized in space on an overlapping grid this system of PDEs can be thought of as a large system of ODEs of the form

$$\frac{d\mathbf{U}}{dt} = \mathcal{F}(t, \mathbf{U}, \mathbf{P})$$

where $\mathbf{U}$ is a vector of all solution values at all grid points. For the purpose of discussing time-stepping methods it is often convenient to think of the pressure as simply a function of $\mathbf{U}$, $\mathbf{P} = \mathcal{P}(\mathbf{U})$ There are also interpolation equations that need to be satisfied but this causes no difficulties.

4 Divergence Damping

The divergence damping term, $C_{d,i}\nabla_h \cdot \mathbf{V}_i$, appears in the pressure equation. In simplified terms, the coefficient C_d is taken proportional to the inverse of the time step, $C_d \sim \frac{1}{\Delta t}$. In practice we have found better results by taking $C_d \sim \frac{\nu}{\Delta x^2}$. For explicit time stepping theses are very similar since the explicit time step restriction is something like $\frac{\nu \Delta t}{\Delta x^2} < C$. To allow for the case $\nu = 0$ we use the minumum grid spacing, $h_{\min}$, instead of ν, if $h_{\min} > \nu$. The size of C_d affects the time step, the stability condition is proportional to $C_d \Delta t$. As a result we do not want C_d to be much larger than $1/\Delta t$, and thus it is limited by $\frac{C_t}{\Delta t}$ where C_t is a constant with default value of 0.25. (Note that we don't actually know the true Δt at this point, it depends on C_d, so we just use a guess).

Here is the actual formula for the divergence damping coefficient:

$$C_{d,i} = \min(\mathcal{D}_i, \frac{C_t}{\Delta t})$$

where

$$\begin{aligned}
\mathcal{D}_i &= C_0 \ \max(\nu, h_{\min}) \ \left(\frac{1}{(\Delta_{0,r_1} x_{1,i})^2} + \frac{1}{(\Delta_{0,r_2} x_{2,i})^2} + \frac{1}{(\Delta_{0,r_3} x_{3,i})^2} \right) \\
\Delta_{0,r_1} x_{m,i} &= \frac{1}{2}(x_{m,i_1+1} - x_{m,i_1-1}) \qquad \text{(undivided second difference)} \\
h_{\min} &= \min_i(\|\Delta_{+,r_1}\mathbf{x}_i\|, \Delta_{+,r_2}\mathbf{x}_i\|, \Delta_{+,r_3}\mathbf{x}_i\|) \qquad \text{(minimum grid spacing)}
\end{aligned}$$

and where $C_0 = 1.$ by default.

5 Artificial Diffusion

OverBlown implements an artificial diffusion based on a second-order undivided difference or a fourth-order undivided difference. The second-order artificial diffusion is

$$\mathbf{d}_{2,i} = (\texttt{ad21} + \texttt{ad22}|\nabla_h \mathbf{V}_i|_1) \sum_{m=1}^{n_d} \Delta_{m+}\Delta_{m-}\mathbf{V}_i \tag{4}$$

while the in the fourth-order one is

$$\mathbf{d}_{4,i} = -(\texttt{ad41} + \texttt{ad42}|\nabla_h \mathbf{V}_i|_1) \sum_{m=1}^{n_d} \Delta_{m+}^2\Delta_{m-}^2\mathbf{V}_i \tag{5}$$

Here $|\nabla_h \mathbf{V}_i|_1$ is the magnitude of the gradient of the velocity and $\Delta_{m\pm}$ are the forward and backward undivided difference operators in direction m

$$\begin{aligned} |\nabla_h \mathbf{V}_i|_1 &= n_d^{-2} \textstyle\sum_{m=1}^{n_d} \sum_{n=1}^{n_d} |D_{m,h} V_{ni}| \\ \Delta_{1+}\mathbf{V}_i &= \mathbf{V}_{i_1+1} - \mathbf{V}_i \\ \Delta_{1-}\mathbf{V}_i &= \mathbf{V}_i - \mathbf{V}_{i_1-1} \\ \Delta_{2+}\mathbf{V}_i &= \mathbf{V}_{i_2+1} - \mathbf{V}_i \\ \Delta_{2-}\mathbf{V}_i &= \mathbf{V}_i - \mathbf{V}_{i_2-1} \quad \text{etc.} \end{aligned}$$

The artificial diffusion is added to the momentum equations

$$\frac{d}{dt}\mathbf{V}_i + (\mathbf{V}_i \cdot \nabla_h)\mathbf{V}_i + \nabla_h P_i - \nu \Delta_h \mathbf{V}_i - \mathbf{f}(\mathbf{x}_i, t) - \mathbf{d}_{m,i} = 0$$

but does not change the pressure equation. Typical choices for the constants $\texttt{ad21} = \texttt{ad41} = 1$ and $\texttt{ad22} = \texttt{ad42} = .5$. These artificial diffusions should not affect the order of accuracy of the method. With the artificial diffusion turned on to a sufficient degree, the real viscosity can be set at low as zero, $\texttt{nu} = 0$.

This form of the artficial diffusion is based on a theoretical result [6][7] that states that the minimum scale, $\lambda_{\min}$, of solutions to the incompressible Navier-Stokes equations is proportional to the square root of the kinematic viscosity divided by the square root of the maximum velocity gradient:

$$\lambda_{\min} \propto \sqrt{\frac{\nu}{|\nabla \mathbf{u}| + c}} \,.$$

This result is valid locally in space so that $|\nabla \mathbf{u}|$ measures the local value of the velocity gradient. The minimum scale measures the size of the smallest eddy or width of the sharpest shear layer as a function of the viscosity and the size of the gradients of $\mathbf{u}$. Scales smaller than the minimum scale are in the exponentially small part of the spectrum.

This result can be used to tell us the smallest value that we can choose for the (artificial) viscosity, ν_A, and still obtain a reasonable numerical solution. We require that the artificial viscosity be large enough so that the smallest (but still significant) features of the flow are resolved on the given mesh. If the local grid spacing is h, then we need

$$h \propto \sqrt{\frac{\nu_A}{|\nabla \mathbf{u}| + c}} \,.$$

This gives

$$\nu_A = (c_1 + c_2 |\nabla \mathbf{u}|) h^2$$

and thus we can choose an artificial diffusion of

$$(c_1 + c_2 |\nabla \mathbf{u}|) h^2 \Delta \mathbf{u}$$

which is just the form (4).

In the fourth-order case we wish to add an artificial diffusion of the form

$$-\nu_A \Delta^2 \mathbf{u}$$

since, as we will see, this will lead to $\nu_A \propto h^4$. In this case, if we consider solutions to the incompressible Navier-Stokes equations with the diffusion term $\nu \Delta \mathbf{u}$ replaced by $-\nu_A \Delta^2 \mathbf{u}$ then the minimum scale would be

$$\lambda_{\min} \propto \left(\frac{\nu_A}{|\nabla \mathbf{u}|} \right)^{1/4}$$

Following the previous argument leads us to choose an artificial diffusion of the form

$$-(c_1 + c_2 |\nabla \mathbf{u}|) h^4 \Delta^2 \mathbf{u}$$

which is just like (5).

6 Boundary Conditions

The boundary conditions for method INS are

$$\text{noSlipWall} = \begin{cases} \mathbf{u} = \mathbf{g} & \text{velocity specified} \\ \nabla \cdot \mathbf{u} = 0 & \text{divergence zero} \end{cases}$$

$$\text{slipWall} = \begin{cases} \mathbf{n} \cdot \mathbf{u} = g & \text{normal velocity specified} \\ \partial_n(\mathbf{t}_m \cdot \mathbf{u}) = 0 & \text{normal derivative of tangential velcity is zero} \\ \nabla \cdot \mathbf{u} = 0 & \text{divergence zero} \end{cases}$$

$$\text{inflowWithVelocityGiven} = \begin{cases} \mathbf{u} = \mathbf{g} & \text{velocity specified} \\ \partial_n p = 0 & \text{normal derivative of the pressure zero.} \end{cases}$$

$$\text{outflow} = \begin{cases} \text{extrapolate } \mathbf{u} \\ \alpha p + \beta \partial_n p = g & \text{mixed derivative of p given.} \end{cases}$$

$$\text{symmetry} = \begin{cases} \mathbf{n} \cdot \mathbf{u}\text{: odd}, \mathbf{t}_m \cdot \mathbf{u}\text{: even} & \text{vector symmetry} \\ \partial_n p = 0 & \text{normal derivative of the pressure zero.} \end{cases}$$

$$\text{dirichletBoundaryCondition} = \begin{cases} \mathbf{u} = \mathbf{g} & \text{velocity specified} \\ p = P & \text{pressure given} \end{cases}$$

7 Boundary conditions for the fourth-order method

Here are the analytic and numerical conditions that we impose at a boundary inorder to determine the values of $\mathbf{u}$ at the two ghost points.

noSlipWall: Analytic boundary conditions

$$\mathbf{u} = \mathbf{u}_B(\mathbf{x}, t)$$

plus numerical boundary conditions

$$\begin{aligned} \mathbf{t}_\mu \cdot \Big\{ \nu\Delta\mathbf{u} - \nabla p - (\mathbf{u}\cdot\nabla)\mathbf{u} - \mathbf{u}_t \Big\} &= 0 \\ \text{Extrapolate } \ \mathbf{t}_\mu \cdot \mathbf{u} &= 0 \\ \nabla \cdot \mathbf{u} &= 0 \\ \partial_n(\nabla \cdot \mathbf{u}) &= 0 \end{aligned}$$

inflowWithVelocityGiven or outflow: Analytic boundary conditions for inflow are

$$\mathbf{u} = \mathbf{u}_I(\mathbf{x}, t) \qquad \text{(inflow)}$$

For outflow the equation is used on the boundary. The numerical boundary conditions are

$$\begin{aligned} \mathbf{t}_\mu \cdot (\mathbf{u}_{nn}) &= 0 \\ \text{Extrapolate } \ \mathbf{t}_\mu \cdot \mathbf{u} &= 0 \\ \nabla \cdot \mathbf{u} &= 0 \\ \partial_n(\nabla \cdot \mathbf{u}) &= 0 \end{aligned}$$

slipWall: Analytic boundary conditions are

$$\mathbf{n} \cdot \mathbf{u} = \mathbf{n} \cdot \mathbf{u}_B$$

The numerical boundary conditions are

$$\begin{aligned} \mathbf{t}_\mu \cdot \Big\{ \nu\Delta\mathbf{u} - \nabla p - (\mathbf{u}\cdot\nabla)\mathbf{u} - \mathbf{u}_t \Big\} &= 0 \quad \text{determines } \mathbf{t}_\mu \cdot \mathbf{u} \text{ on the boundary} \\ \mathbf{t}_\mu \cdot (\mathbf{u}_n) &= 0 \\ \mathbf{t}_\mu \cdot (\mathbf{u}_{nnn}) &= 0 \\ \nabla \cdot \mathbf{u} &= 0 \\ \partial_n(\nabla \cdot \mathbf{u}) &= 0 \end{aligned}$$

Discretizing the Boundary conditions: For the purposes of this discussion assume that the boundary condition for $\mathbf{u}$ is of the form $\mathbf{u}(\mathbf{x}, t) = \mathbf{u}_B(\mathbf{x}, t)$ for $\mathbf{x} \in \partial\Omega$. More general boundary conditions on $\mathbf{u}$ and p, such as extrapolation conditions, can also be dealt with although some of the details of implementation may vary. At a boundary the following conditions are applied

$$\left.\begin{array}{rcl} \mathbf{U}_i - \mathbf{u}_B(\mathbf{x}_i) &=& 0 \\ \nabla_4 \cdot \mathbf{U}_i &=& 0 \\ D_{4n}(\nabla_4 \cdot \mathbf{U}_i) &=& 0 \\ \frac{d}{dt}\mathbf{U}_i + (\mathbf{U}_i \cdot \nabla_4)\mathbf{U}_i + \nabla_4 P_i - \nu \Delta_4 \mathbf{U}_i - \mathbf{f}_i &=& 0 \\ \Delta_4 P_i + \sum_{m=1}^{n_d} \nabla_4 U_{m,i} \cdot D_{4x_m}\mathbf{U}_i - \nabla_4 \cdot \mathbf{f}_i &=& 0 \end{array}\right\} \quad \text{for } i \in \text{Boundary}$$

$$\left.\begin{array}{rcl} \mathbf{t}_\mu \cdot D_{+m}^4 \mathbf{U}_i &=& 0 \\ D_{+m}^4 P_i &=& 0 \end{array}\right\} \quad \text{for } i \in \text{2nd fictitious line}$$

where $\mathbf{t}_\mu$, $\mu = 1, n_d - 1$ are linearly independent vectors that are tangent to the boundary. In the extrapolation conditions either D_{+m} or D_{-m} should be chosen, as appropriate. Thus at each point along the boundary there are 12 equations for the 12 unknowns $(\mathbf{U}_i, P_i)$ located on the boundary and the 2 lines of fictitious points. Note that two of the numerical boundary conditions couple the pressure and velocity. In order to advance the velocity with an explicit time stepping method it convenient to decouple the solution of the pressure equation from the solution of the velocity. A procedure to accomplish this is described in the next section on time stepping.

Edges and Vertices: An important special case concerns obtaining solution values at points that lie near edges and vertices of grids (or corners of grids in 2D). Define a *boundary edge* to be the edge that is formed at the intersection of adjacent faces of the unit cube where both faces are boundaries of the computational domain. Along a boundary edge, values of the solution are required at the fictitious points in the region exterior to both boundary faces. For example, suppose that the edge defined by $i_1 = n_{1,a}$, $i_2 = n_{2,a}$ and $i_3 = n_{3,a}, \ldots, n_{3,b}$ is a boundary edge. Values must be determined at the exterior points $i = (n_{1,a} + m, n_{2,a} + n, i_3)$ for $m, n = -2, -1$.

Here we derive a more accurate formula than was in my paper. These expressions will be exact for polynomials of degree 4. By Taylor series,

$$u(r_1, r_2) = u(0,0) + \mathcal{D}_1(r_1, r_2) + \mathcal{D}_2(r_1, r_2) + \mathcal{D}_3(r_1, r_2) + \mathcal{D}_4(r_1, r_2) + O(|\mathbf{r}|^6)$$

where

$$\begin{aligned} \mathcal{D}_1(r_1, r_2) &= (r_1 \partial_{r_1} + r_2 \partial_{r_2}) u(0,0) \\ \mathcal{D}_2(r_1, r_2) &= \frac{1}{2}({r_1}^2 \partial_{r_1}^2 + {r_2}^2 \partial_{r_2}^2 + 2 r_1 r_2 \partial_{r_1} \partial_{r_2}) u(0,0) \\ \mathcal{D}_3(r_1, r_2) &= \frac{1}{3!}({r_1}^3 \partial_r^3 + {r_2}^3 \partial_{r_2}^3 + 3{r_1}^2 r_2 \partial_{r_1}^2 \partial_{r_2} + 3 r_1 {r_2}^2 \partial_{r_1} \partial_{r_2}^2) u(0,0) \end{aligned}$$

We also have

$$u(-r_1, -r_2) = 2u(0,0) - u(r_1, r_2) + 2\mathcal{D}_2(r_1, r_2) + 2\mathcal{D}_4(r_1, r_2) + O(|\mathbf{r}|^6) \tag{6}$$

$$u(2r_1, 2r_2) = u(0,0) + 2\mathcal{D}_1(r_1, r_2) + 4\mathcal{D}_2(r_1, r_2) + 8\mathcal{D}_3(r_1, r_2) + 16\mathcal{D}_4(r_1, r_2) + O(|\mathbf{r}|^6) \tag{7}$$

$$8u(r_1, r_2) - u(2r_1, 2r_2) = 7u(0,0) + 6\mathcal{D}_1(r_1, r_2) + 4\mathcal{D}_2(r_1, r_2) - 8\mathcal{D}_4(r_1, r_2) + O(|\mathbf{r}|^6) \tag{8}$$

From equation (8) we can solve for $\mathcal{D}_4(r_1, r_2)$,

$$\mathcal{D}_4(r_1, r_2) = \frac{7}{8}u(0,0) - u(r_1, r_2) + \frac{1}{8}u(2r_1, 2r_2) + \frac{3}{4}\mathcal{D}_1(r_1, r_2) + \frac{1}{2}\mathcal{D}_2(r_1, r_2) + O(|r|^5 + |s|^5)$$

and substitute into equation (6)

$$u(-r_1,-r_2) = \frac{15}{4}u(0,0) - 3u(r_1,r_2) + \frac{1}{4}u(2r_1,2r_2) + \frac{3}{2}\mathcal{D}_1(r_1,r_2) + 3\mathcal{D}_2(r_1,r_2) + O(|\mathbf{r}|^6) \tag{9}$$

We will use this last equation to determine u at the first corner ghost point, $\mathbf{U}_{-1,-1,i_3}$. Proceeding in a similar way it follows that

$$u(-2r_1,-r_2) = \frac{15}{4}u(0,0) - 3u(2r_1,r_2) + \frac{1}{4}u(4r_1,2r_2) + \frac{3}{2}\mathcal{D}_1(2r_1,r_2) + 3\mathcal{D}_2(2r_1,r_2) + O(|\mathbf{r}|^6)O(|\mathbf{r}|^6) \tag{10}$$

$$u(-r_1,-2r_2) = \frac{15}{4}u(0,0) - 3u(r_1,2r_2) + \frac{1}{4}u(2r_1,4r_2) + \frac{3}{2}\mathcal{D}_1(r_1,2r_2) + 3\mathcal{D}_2(r_1,2r_2) + O(|\mathbf{r}|^6)O(|\mathbf{r}|^6) \tag{11}$$

$$u(-2r_1,-2r_2) = 30u(0,0) - 32u(r_1,r_2) + 3u(2r_1,2r_2) + 24\mathcal{D}_1(r_1,r_2) + 24\mathcal{D}_2(r_1,r_2) + O(|\mathbf{r}|^6)O(|\mathbf{r}|^6) \tag{12}$$

from which we will determine the ghost points values at $\mathbf{U}_{-2,-2,i_3}$, $\mathbf{U}_{-2,-1,i_3}$ and $\mathbf{U}_{-1,-2,i_3}$. By symmetry we obtain formulae for ghost points outside all other edges in three-dimensions, $\mathbf{U}_{-2:-1,i_2,-2:-1}$, $\mathbf{U}_{i_1,-2:-1,-2:-1}$. For ghost points outside the vertices in three-dimensions we have

$$u(-r_1,-r_2,-r_3) = \frac{15}{4}u(0,0) - 3u(r_1,r_2,r_3) + \frac{1}{4}u(2r_1,2r_2,2r_3) \tag{13}$$

$$+ \frac{3}{2}\mathcal{D}_1(r_1,r_2,r_3) + 3\mathcal{D}_2(r_1,r_2,r_3) + O(|\mathbf{r}|^6)O(|\mathbf{r}|^6) \tag{14}$$

where

$$\begin{aligned}
\mathcal{D}_1(r_1,r_2,r_3) &= \Big(r_1\partial_{r_1} + r_2\partial_{r_2} + r_3\partial_{r_3}\Big)u(0,0,0) \\
\mathcal{D}_2(r_1,r_2,r_3) &= \frac{1}{2}\Big({r_1}^2\partial^2_{r_1} + {r_2}^2\partial^2_{r_2} + {r_3}^2\partial^2_{r_3} + 2r_1r_2\partial_{r_1}\partial_{r_2} + 2r_1r_3\partial_{r_1}\partial_{r_3} + 2r_2r_3\partial_{r_2}\partial_{r_3}\Big)u(0,0,0) \\
\mathcal{D}_3(r_1,r_2,r_3) &= \frac{1}{3!}\sum_{m_1=1}^{3}\sum_{m_2=1}^{3}\sum_{m_3=1}^{3} r_{m_1}r_{m_2}r_{m_3}\partial_{m_1}\partial_{m_2}\partial_{m_3}u(0,0,0) \\
&= \frac{1}{3!}\Big({r_1}^3\partial^3_{r} + {r_2}^3\partial^3_{r_2} + {r_3}^3\partial^3_{r_3} + 3{r_1}^2r_2\partial^2_{r_1}\partial_{r_2} + 3r_1{r_2}^2\partial_{r_1}\partial^2_{r_2} \\
&+ 3{r_1}^2r_3\partial^2_{r_1}\partial_{r_3} + 3r_1{r_3}^2\partial_{r_1}\partial^2_{r_3} + 3{r_2}^2r_3\partial^2_{r_2}\partial_{r_3} + 3r_2{r_3}^2\partial_{r_2}\partial^2_{r_3} + 6r_1r_2r_3\partial_{r_1}\partial_{r_2}\partial_{r_3}\Big)u(0,0,0)
\end{aligned}$$

In order to evaluate the formulae (9,10,11, 12,14) we need to evaluate the derivatives appearing in $\mathcal{D}_1$ and $\mathcal{D}_2$. All the non-mixed derivatives $\partial^m u(0,0)/\partial^m_{r_n}$, $m = 1,2$, can be evaluated using the boundary values since these are all tangential derivatives. The second-order mixed derivative term such as $u_{r_1r_2}$ requires a bit more work. In two-dimensions we evaluate this term by taking the parametric derivatives of the divergence, $\partial_{r_m}\nabla\cdot\mathbf{u} = 0$. Since

$$\nabla\cdot\mathbf{u} = \sum_{n=1}^{2}(\partial_x r_n)\,u_{r_n} \;+\; \sum_{n=1}^{2}(\partial_y r_n)\,v_{r_n}$$

then $\partial_{r_m}\nabla\cdot\mathbf{u} = 0$ gives

$$\begin{aligned}
(\partial_x r_2)\,u_{r_1r_2} + (\partial_y r_2)\,v_{r_1r_2} =& (\partial_x r_2)_{r_1}\,u_{r_2} + (\partial_x r_1)\,u_{r_1r_1} + (\partial_x r_1)_{r_1}\,\partial_{r_1}u + \\
& (\partial_y r_2)_{r_1}\,v_{r_2} + (\partial_y r_1)\,v_{r_1r_1} + (\partial_y r_1)_{r_1}\,\partial_{r_1}v \\
(\partial_x r_1)\,u_{r_1r_2} + (\partial_y r_1)\,v_{r_1r_2} =& (\partial_x r_2)\,u_{r_2r_2} + (\partial_x r_2)_{r_2}\,\partial_{r_2}u ...
\end{aligned}$$

These last two equations can be solved for $u_{r_1 r_2}$ and $v_{r_1 r_2}$ in terms of known tangential derivatives.

In three dimensions taking the two parametric derivatives of the divergence,

$$(\partial_x r_2)\, u_{r_1 r_2} + (\partial_y r_2)\, v_{r_1 r_2} + (\partial_z r_2)\, w_{r_1 r_2} = ... \tag{15}$$
$$(\partial_x r_1)\, u_{r_1 r_2} + (\partial_y r_1)\, v_{r_1 r_2} + (\partial_z r_1)\, w_{r_1 r_2} = ... \tag{16}$$

gives only two equations for the three unknowns, $u_{r_1 r_2}, v_{r_1 r_2}$ and $w_{r_1 r_2}$. We therefore add an extra condition by extrapolating the tangential component of the velocity,

$$\mathbf{t}_3 \cdot D^6_{+,1,2} \mathbf{U}_{i_1-1,i_2-1,i_3} = 0 \tag{17}$$

Solve the last equation for $\mathbf{t}_3 \cdot \mathbf{U}_{i_1-1,i_2-1,i_3}$ gives

$$\mathbf{t}_3 \cdot \mathbf{U}_{i_1-1,i_2-1,i_3} = \mathcal{E}^6_{+,1,2} \mathbf{t}_3 \cdot \mathbf{U}_{i_1-1,i_2-1,i_3} \tag{18}$$

where we have introduced the operator $\mathcal{E}^6_{+,1,2}$. By substituting this last equation (18) into $\mathbf{t}_3 \cdot$ equation (9),

$$\mathbf{t}_3 \cdot \mathbf{u}(-r_1,-r_2) = \mathbf{t}_3 \cdot \Big(\frac{15}{4}\mathbf{u}(0,0) - 3\mathbf{u}(r_1,r_2) + \frac{1}{4}\mathbf{u}(2r_1,2r_2) + \frac{3}{2}\mathcal{D}_1(r_1,r_2)\mathbf{u}(0) + 3\mathcal{D}_2(r_1,r_2)\mathbf{u}(0) \Big) + O(|\mathbf{r}|^6)$$

we can eliminate $\mathbf{t}_3 \cdot \mathbf{U}_{i_1-1,i_2-1,i_3}$ and obtain an equation for $\mathbf{t}_3 \cdot \mathbf{u}_{r_1 r_2}$ in terms of known quanitites,

$$\begin{aligned} \mathbf{t}_3 \cdot \left(\mathcal{E}^6_{+,1,2} \mathbf{U}_{i_1-1,i_2-1,i_3} \right) = \mathbf{t}_3 \cdot \Big(& \frac{15}{4}\mathbf{u}(0,0) - 3\mathbf{u}(r_1,r_2) + \frac{1}{4}\mathbf{u}(2r_1,2r_2) + \frac{3}{2}\mathcal{D}_1(r_1,r_2)\mathbf{u}(0) \\ & + \frac{3}{2}({r_1}^2\partial^2_{r_1} + {r_2}^2\partial^2_{r_2} + {r_3}^2\partial^2_{r_3} + 2r_1r_2\partial_{r_1}\partial_{r_2} + 2r_1r_3\partial_{r_1}\partial_{r_3} + 2r_2r_3\partial_{r_2}\partial_{r_3})\mathbf{u}(0,0) \Big) \end{aligned}$$

or re-written as

$$\begin{aligned} \mathbf{t}_3 \cdot \mathbf{u}_{r_1 r_2}(0,0) = \frac{1}{3}\Big\{ & \mathbf{t}_3 \cdot \left(\mathcal{E}^6_{+,1,2} \mathbf{U}_{i_1-1,i_2-1,i_3} \right) - \mathbf{t}_3 \cdot \Big(\frac{15}{4}\mathbf{u}(0,0) - 3\mathbf{u}(r_1,r_2) + \frac{1}{4}\mathbf{u}(2r_1,2r_2) + \frac{3}{2}\mathcal{D}_1(r_1,r_2)\mathbf{u}(0) \\ & + \frac{3}{2}({r_1}^2\partial^2_{r_1} + {r_2}^2\partial^2_{r_2} + {r_3}^2\partial^2_{r_3} + 2r_1r_3\partial_{r_1}\partial_{r_3} + 2r_2r_3\partial_{r_2}\partial_{r_3})\mathbf{u}(0,0) \Big\} \end{aligned} \tag{19}$$

To summarize we solve equations (15,16,19) for the three unknowns $u_{r_1 r_2}, v_{r_1 r_2}$ and $w_{r_1 r_2}$.

Solving the numerical boundary equations: The numerical boundary conditions (**??**) define the values of $\mathbf{U}$ on two lines of fictitious points in terms of values of the velocity on the boundary and the interior. The equations couple the unknowns in the tangential direction to the boundary so that in principle a system of equations for all boundary points must be solved. However, when the grid is nearly orthogonal to the boundary there is a much more efficient way to solve the boundary conditions. The first step in the algorithm is to solve for the tangential components of the velocity from

$$\left.\begin{aligned} \mathbf{U}_i(t) - \mathbf{u}_B(\mathbf{x}_i,t) &= 0 \\ \mathbf{t}_\mu \cdot \left\{ \tfrac{d}{dt}\mathbf{U}_i(t) + (\mathbf{U}_i(t)\cdot\nabla_4)\mathbf{U}_i(t) + \nabla_4 P^*(t) - \nu\Delta_4\mathbf{U}_i(t) - \mathbf{f} \right\} &= 0 \end{aligned}\right\} \quad \text{for } i \in \text{Boundary}$$
$$\mathbf{t}_\mu \cdot D^6_{+m}(\mathbf{U}_i(t)) = 0 \;\Big\} \quad \text{for } i \in \text{Second fictitious line}$$

If the grid is orthogonal to the boundary then the discrete Laplacian applied at boundary will not have any mixed derivative terms. Therefore the only fictitious points appearing in the equation applied at the the boundary point (i_1, i_2, i_3) will be the two points $(i_1, i_2, i_3 - n)$ $n = 1, 2$ (here we assume that i_3 is in the normal direction to the boundary). Thus for each point on the boundary (i_1, i_2, i_3) the values of $\mathbf{t}_\mu \cdot \mathbf{u}$ can be determined at the fictitious points $(i_1, i_2, i_3 - 1)$ and $(i_1, i_2, i_3 - 2)$. There is no coupling between adjacent boundary points so no large system of equations need be solved. The tangential components of the velocity are determined for all fictitious points on the entire boundary. The second step is to determine the the normal component of the velocity at the fictitious points from

$$\left.\begin{array}{rcl} \mathbf{U}_i(t) - \mathbf{u}_B(\mathbf{x}_i, t) & = & 0 \\ \nabla_4 \cdot \mathbf{U}_i(t) & = & 0 \\ D_{4n}(\nabla_4 \cdot \mathbf{U}_i(t)) & = & 0 \end{array}\right\} \qquad \text{for } i \in \text{Boundary}$$

If the grid is orthogonal to the boundary then the divergence on the boundary can be written in the form

$$\nabla \cdot \mathbf{u} = \frac{1}{e_1 e_2 e_3} \left\{ \frac{\partial}{\partial n}(e_2 e_3 \mathbf{n} \cdot \mathbf{u}) + \frac{\partial}{\partial t_1}(e_1 e_3 \mathbf{t}_1 \cdot \mathbf{u}) + \frac{\partial}{\partial t_2}(e_1 e_2 \mathbf{t}_2 \cdot \mathbf{u}) \right\}$$

where the e_m are functions of $\partial \mathbf{x}/\partial \mathbf{r}$. Note that only normal derivatives of $\mathbf{n} \cdot \mathbf{u}$ appear in the expression for the divergence. Thus, at a boundary point, (i_1, i_2, i_3), the stencil for $\nabla_4 \cdot \mathbf{U}$ will only involve the fictitious points at $(i_1, i_2, i_3 - n)$, $n = 1, 2$. Similarly, the stencil for $D_{4n}(\nabla_4 \cdot \mathbf{U})$ at a boundary will only involve the fictitious points at $(i_1, i_2, i_3 - n)$, $n = 1, 2$. Thus there is no coupling between adjacent boundary points and the unknown values for $\mathbf{n} \cdot \mathbf{u}$ can be easily determined. Note that the equations for $D_{4n}(\nabla_4 \cdot \mathbf{U})$ will couple values for $\mathbf{t}_\mu \cdot \mathbf{u}$ at fictitious points along the boundary but these values have already been determined in the first step.

In practice the boundary conditions are solved in a correction mode – some initial guess is assumed for the values at the fictitious points and a correction is computed. If the grid is orthogonal or nearly orthogonal to the boundary then the first correction will give an accurate answer to the boundary conditions. If the grid is not orthogonal to the boundary then the solution procedure can repeated one or more times until a desired accuracy is achieved. This iteration should converge quickly provided that the grid is not overly skewed.

8 Turbulence models

The typical RANS model for the incompressible Navier-Stokes equations which uses the Boussinesq eddy viscosity approximation is

$$\partial_t u_i + u_k \partial_{x_k} u_i + \partial_{x_i} p = \partial_{x_k}\Big((\nu + \nu_T)(\partial_{x_k} u_i + \partial_{x_i} u_k) \Big)$$

where ν_T is the turbulent eddy viscosity.

8.1 Spalart-Allmaras turbulence model

Spalart-Allmaras **one equation model**

$$\begin{aligned}
\nu_T &= \tilde{\nu} f_{v1} \\
\partial_t \tilde{\nu} + U_j \partial_j \tilde{\nu} &= c_{b1} \tilde{S} \tilde{\nu} - c_{w1} f_w (\tilde{\nu}/d)^2 + \frac{1}{\sigma}\Big[\partial_k [(\nu + \tilde{\nu}) \partial_k \tilde{\nu}] + c_{b2} \partial_k \tilde{\nu} \partial_k \tilde{\nu} \Big] \\
c_{b1} &= .1355, c_{b2} = .622, c_{v1} = 7.1, \sigma = 2/3 \\
c_{w1} &= \frac{c_{b1}}{\kappa^2} + \frac{(1 + c_{b2})}{\sigma}, \quad c_{w2} = 0.3, \quad c_{w3} = 2, \quad \kappa = .41 \\
f_{v1} &= \frac{\chi^3}{\chi^3 + c_{v1}^3}, \quad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}}, f_w = g \left[\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6} \right]^{1/6} \\
\chi = \frac{\tilde{\nu}}{\nu}, \quad g &= r + c_{w2}(r^6 - r), \qquad r = \frac{\tilde{\nu}}{\tilde{S} \kappa^2 d^2} \\
\tilde{S} &= S + \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2}, \qquad S = \sqrt{2 \Omega_{ij} \Omega_{ij}} \\
\Omega_{ij} &= (1/2)(\partial_i U_j - \partial_j U_i) \qquad \text{rotation tensor}
\end{aligned}$$

Depends on d, the distance to the nearest surface.

Notes f_{v2} can be positive or negative but is bounded from above by 1 and below by ??

$$\begin{aligned}
f_{v2} &= 1 - \frac{\chi}{1 + \chi f_{v1}} \\
&= 1 - \frac{1}{\chi^{-1} + 1/[1 + (c_{v1}/\chi)^{-3}]} \\
&\to 0 \qquad \text{as } \chi \to \infty \\
&\to 1 \qquad \text{as } \chi \to 0 \\
&\approx 1 - \frac{1}{(1/7) + (1/2)} = -3/4 \qquad \text{when } \chi = c_{v1}
\end{aligned}$$

Since f_{v2} can be negative, so can r and g.

On a rectangular grid this is discretized as

$$\begin{aligned}
\partial_t \tilde{\nu}_{\mathbf{i}} + U_{\mathbf{i}} D_{0x} U_{\mathbf{i}} + V_{\mathbf{i}} D_{0y} U_{\mathbf{i}} = c_{b1} \tilde{S}_{\mathbf{i}} \tilde{\nu}_{\mathbf{i}} &- c_{w1} f_w (\tilde{\nu}/d)^2 \\
&+ \frac{1}{\sigma} D_{+x}[(\nu + \tilde{\nu}_{i_1 - \frac{1}{2}}) D_{-x} \tilde{\nu}_{\mathbf{i}} + D_{+y}[(\nu + \tilde{\nu}_{i_2 - \frac{1}{2}}) D_{-y} \tilde{\nu}_{\mathbf{i}}] \\
&+ c_{b2} \Big\{ (D_{0x} \tilde{\nu})^2 + (D_{0y} \tilde{\nu})^2 \Big\}
\end{aligned}$$

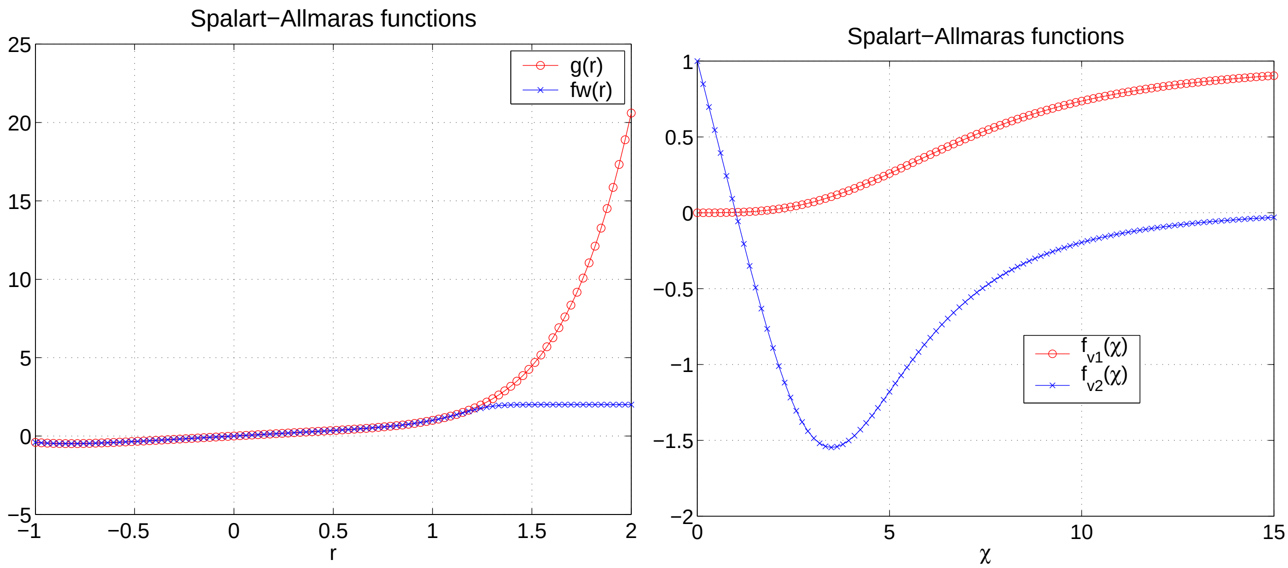


Figure 1: Behaviour of the SpalartConvergence fudge functions

On curvilinear grids we use the conservative form of the second order term

$$\nabla\cdot(a\nabla\phi)=\frac{1}{J}\left\{\frac{\partial}{\partial r_1}\left(A^{11}\frac{\partial\phi}{\partial r_1}\right)+\frac{\partial}{\partial r_2}\left(A^{22}\frac{\partial\phi}{\partial r_2}\right)+\frac{\partial}{\partial r_1}\left(A^{12}\frac{\partial\phi}{\partial r_2}\right)+\frac{\partial}{\partial r_2}\left(A^{21}\frac{\partial\phi}{\partial r_1}\right)\right\}$$

where

$$\begin{aligned}
A^{11} &= aJ\left[\frac{\partial r_1}{\partial x_1}^2+\frac{\partial r_1}{\partial x_2}^2\right]\\
A^{22} &= aJ\left[\frac{\partial r_2}{\partial x_1}^2+\frac{\partial r_2}{\partial x_2}^2\right]\\
A^{12} &= aJ\left[\frac{\partial r_1}{\partial x_1}\frac{\partial r_2}{\partial x_1}+\frac{\partial r_1}{\partial x_2}\frac{\partial r_2}{\partial x_2}\right]
\end{aligned}$$

A **second-order accurate** compact discretization to this expression is

$$\nabla\cdot(a\nabla\phi)\approx\frac{1}{J}\left\{D_{+r_1}\left(A^{11}_{i_1-\frac{1}{2}}D_{-r_1}\phi\right)+D_{+r_2}\left(A^{22}_{i_2-\frac{1}{2}}D_{-r_2}\phi\right)+D_{0r_1}\left(A^{12}D_{0r_2}\phi\right)+D_{0r_2}\left(A^{21}D_{0r_1}\phi\right)\right\}$$

where we can define the cell average values for A^{mn} by

$$\begin{aligned}
A^{11}_{i_1-\frac{1}{2}} &\approx \frac{1}{2}(A^{11}_{i_1}+A^{11}_{i_1-1})\\
A^{22}_{i_2-\frac{1}{2}} &\approx \frac{1}{2}(A^{22}_{i_2}+A^{22}_{i_2-1})
\end{aligned}$$

8.2 $k - \epsilon$ turbulence model

Here is the $k - \epsilon$ model

$$\begin{aligned} \nu_T &= C_\mu k^2/\epsilon \\ \partial_t k + U_j \partial_j k &= \tau_{ij}\partial_j U_i - \epsilon + \partial_j[(\nu + \nu_T/\sigma_k)\partial_j k] \\ \partial_t \epsilon + U_j \partial_j \epsilon &= C_{\epsilon 1}\frac{\epsilon}{k}\tau_{ij}\partial_j U_i - C_{\epsilon 2}\epsilon^2/k + \partial_j[(\nu + \nu_T/\sigma_\epsilon)\partial_j \epsilon] \\ C_{\epsilon 1} = 1.44, \quad C_{\epsilon 2} &= 1.92, \quad C_\mu = .09, \quad \sigma_k = 1, \quad \sigma_\epsilon = 1.3 \end{aligned}$$

The production term is $P = \tau_{ij}\partial_j U_i$

$$\begin{aligned} P &= \tau_{ij}\partial_j U_i \\ &= \nu_T(\partial_j U_i + \partial_i U_j)\partial_j U_i \\ &= \frac{\nu_T}{2}(\partial_j U_i + \partial_i U_j)(\partial_j U_i + \partial_i U_j) \\ &= \frac{\nu_T}{2}\Big((2u_x)^2 + (2v_y)^2 + (2w_z)^2 + 2(u_y + v_x)^2 + 2(u_z + w_x)^2 + 2(v_z + w_y)^2\Big) \\ &= \nu_T\Big(2(u_x^2 + v_y^2 + w_z^2) + (u_y + v_x)^2 + (u_z + w_x)^2 + (v_z + w_y)^2\Big) \end{aligned}$$

8.3 Diffusion Operator

When a turbulence model is added to the incompressible Navier-Stokes equation the diffusion operator usually takes the form of

$$\mathcal{D}_i = \sum_j \partial_{x_j}\Big(\nu_T(\partial_{x_i} u_j + \partial_{x_j} u_i)\Big)$$

where we will write ν_T instead of $\nu + \nu_T$ in this section. In particular

$$\begin{aligned} \mathcal{D}_u &= \partial_x(2\nu_T u_x) + \partial_y(\nu_T u_y) + \partial_z(\nu_T u_z) + \partial_y(\nu_T v_x) + \partial_z(\nu_T w_x) \\ \mathcal{D}_v &= \partial_x(\nu_T v_x) + \partial_y(2\nu_T v_y) + \partial_z(\nu_T v_z) + \partial_x(\nu_T u_y) + \partial_z(\nu_T w_y) \\ \mathcal{D}_w &= \partial_x(\nu_T w_x) + \partial_y(\nu_T w_y) + \partial_z(2\nu_T w_z) + \partial_y(\nu_T v_z) + \partial_x(\nu_T u_z) \end{aligned}$$

We can write these in a more “symmetric” form as follows. Since $u_x + v_y + w_z = 0$, it follows that

$$\partial_x(\nu_T u_x) = -\partial_x(\nu_T v_y) - \partial_x(\nu_T w_z)$$

and thus

$$\mathcal{D}_u = \partial_x(\nu_T u_x) + \partial_y(\nu_T u_y) + \partial_z(\nu_T u_z) + \partial_y(\nu_T v_x) - \partial_x(\nu_T v_y) + \partial_z(\nu_T w_x) - \partial_x(\nu_T w_z)$$

Therefore the diffusion operator can be written in a form where the principle part is the same for all components,

$$\begin{aligned} \mathcal{D}_u &= \nabla\cdot(\nu_T \nabla u) + \partial_y(\nu_T v_x) - \partial_x(\nu_T v_y) + \partial_z(\nu_T w_x) - \partial_x(\nu_T w_z) \\ \mathcal{D}_v &= \nabla\cdot(\nu_T \nabla v) + \partial_z(\nu_T w_y) - \partial_y(\nu_T w_z) + \partial_x(\nu_T u_y) - \partial_y(\nu_T u_x) \\ \mathcal{D}_w &= \nabla\cdot(\nu_T \nabla w) + \partial_x(\nu_T u_z) - \partial_z(\nu_T u_x) + \partial_y(\nu_T v_z) - \partial_z(\nu_T v_y) \end{aligned}$$

Note that the highest order derivatives cancel in the last four terms in these expressions,

$$\begin{aligned}
\mathcal{D}_u &= \nabla\cdot(\nu_T\nabla u) + \partial_y(\nu_T)v_x - \partial_x(\nu_T)v_y + \partial_z(\nu_T)w_x - \partial_x(\nu_T)w_z \\
\mathcal{D}_v &= \nabla\cdot(\nu_T\nabla v) + \partial_z(\nu_T)w_y - \partial_y(\nu_T)w_z + \partial_x(\nu_T)u_y - \partial_y(\nu_T)u_x \\
\mathcal{D}_w &= \nabla\cdot(\nu_T\nabla w) + \partial_x(\nu_T)u_z - \partial_z(\nu_T)u_x + \partial_y(\nu_T)v_z - \partial_z(\nu_T)v_y
\end{aligned}$$

8.4 Revised pressure equation

When a turbulence model is added to the incompressible Navier-Stokes equation the diffusion operator usually takes the form of

$$\mathcal{D}_i = \sum_j \partial_{x_j}\Big(\nu_T(\partial_{x_i}u_j + \partial_{x_j}u_i)\Big)$$

The pressure equation is derived from taking the divergence of the momentum equations,

$$\partial_t\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p = \mathcal{D}$$

and using $\nabla\cdot\mathbf{u} = 0$ to give

$$\Delta p = -\nabla\mathbf{u} : \nabla\mathbf{u} + \nabla\cdot\mathcal{D}$$

For a constant viscosity, the last term on the right hand side is zero. When the viscosity is not constant we need to include the divergence of the diffusion operator in the equation for the pressure. This takes the form

$$\begin{aligned}
\nabla\cdot\mathcal{D} &= \sum_i\sum_j \partial_{x_i}\partial_{x_j}\Big[\nu_T\Big(\partial_{x_j}u_i + \partial_{x_i}u_j\Big)\Big] \\
&= \sum_j \partial_{x_j}\Big[\sum_i \partial_{x_i}\nu_T\,\partial_{x_j}u_i\Big] + \sum_i \partial_{x_i}\Big[\sum_j \partial_{x_j}\nu_T\,\partial_{x_i}u_j\Big] \\
&= 2\,\sum_i \partial_{x_i}\Big[\sum_j\,\partial_{x_j}\nu_T\,\partial_{x_i}u_j\Big]
\end{aligned}$$

where we have again used $\nabla\cdot\mathbf{u} = 0$.

In two dimensions this takes the form

$$\begin{aligned}
\nabla\cdot\mathcal{D}^{(2d)} &= 2\,\Big[\partial_x\Big(\partial_x\nu_T\,\partial_x u + \partial_y\nu_T\,\partial_x v\Big) + \partial_y\Big(\partial_x\nu_T\,\partial_y u + \partial_y\nu_T\,\partial_y v\Big)\Big] \\
&= 2\,\Big[\partial_x\nu_T\,\Delta u + \partial_x^2\nu_T\,\partial_x u + \partial_x\partial_y\nu_T\,\partial_y u \\
&\quad + \partial_y\nu_T\,\Delta v + \partial_x\partial_y\nu_T\,\partial_x v + \partial_y^2\nu_T\,\partial_y v\Big]
\end{aligned}$$

In three dimensions,

$$\begin{aligned}
\nabla\cdot\mathcal{D}^{(3d)} &= 2\,\Big[\partial_x\Big(\partial_x\nu_T\,\partial_x u + \partial_y\nu_T\,\partial_x v + \partial_z\nu_T\,\partial_x w\Big) \\
&\quad + \partial_y\Big(\partial_x\nu_T\,\partial_y u + \partial_y\nu_T\,\partial_y v + \partial_z\nu_T\,\partial_y w\Big) \\
&\quad + \partial_z\Big(\partial_x\nu_T\,\partial_z u + \partial_y\nu_T\,\partial_z v + \partial_z\nu_T\,\partial_z w\Big)\Big] \\
&= 2\,\Big[\partial_x\nu_T\,\Delta u + \partial_x^2\nu_T\,\partial_x u + \partial_x\partial_y\nu_T\,\partial_y u + \partial_x\partial_z\nu_T\,\partial_z u \\
&\quad + \partial_y\nu_T\,\Delta v + \partial_x\partial_y\nu_T\,\partial_x v + \partial_y^2\nu_T\,\partial_y v + \partial_y\partial_z\nu_T\,\partial_z v \\
&\quad + \partial_z\nu_T\,\Delta w + \partial_x\partial_z\nu_T\,\partial_x w + \partial_y\partial_z\nu_T\,\partial_y w + \partial_z^2\nu_T\,\partial_z w\Big]
\end{aligned}$$

The addition of artificial dissipation also changes the pressure equation. The second-order artificial dissipation is

$$\mathbf{d}_{2,i} = (\mathtt{ad21} + \mathtt{ad22}|\nabla_h \mathbf{V}_i|_1) \sum_{m=1}^{n_d} \Delta_{m+}\Delta_{m-}\mathbf{V}_i \tag{20}$$

while the the fourth-order one is

$$\mathbf{d}_{4,i} = -\,(\mathtt{ad41} + \mathtt{ad42}|\nabla_h \mathbf{V}_i|_1) \sum_{m=1}^{n_d} \Delta_{m+}^2\Delta_{m-}^2\mathbf{V}_i \tag{21}$$

The artificial dissipation is of the form

$$\mathbf{d} = \Big[\alpha_0 + \alpha_1 \mathcal{G}(\nabla \mathbf{u})\Big] \sum_{m=1}^{n_d} \Delta_{m+}^p\Delta_{m-}^p\mathbf{u}$$
$$\mathcal{G}(\nabla \mathbf{u}) = |u_x| + |u_y| + |v_x| + |v_y| + \dots$$

Taking the divergence of this expression results in

$$\nabla\cdot\mathbf{d} = \alpha_1\Big[\mathcal{G}_x \sum_m \Delta_{m+}^p\Delta_{m-}^p U + \mathcal{G}_y \sum_m \Delta_{m+}^p\Delta_{m-}^p V\Big]$$

where

$$\mathcal{G}_x = \mathrm{sgn}(u_x)\, u_{xx} + \mathrm{sgn}(u_y)\, u_{xy} + \mathrm{sgn}(v_x)\, v_{xx} + \mathrm{sgn}(v_y)\, v_{xy} + \dots$$

and $\mathrm{sgn}(x)$ is $+1$, -1 or 0 for $x > 0$, $x < 0$ or $x = 0$.

8.4.1 Revised pressure boundary condition

The *pressure boundary condition* also is changed to include the new diffusion operator:

$$\partial_n p = \mathbf{n}\cdot\Big\{-\partial_t\mathbf{u} - (\mathbf{u}\cdot\nabla)\mathbf{u} + \mathcal{D}\Big\}$$

We would like to write the diffusion operator in a way similiar to *curl-curl* form,

$$\Delta\mathbf{u} = -\nabla\times\nabla\times\mathbf{u} + \nabla(\nabla\cdot\mathbf{u}),$$

used in the INS equations. Expanding the expression for $\mathcal{D}_u$ gives

$$\begin{aligned}
\mathcal{D}_u &= \partial_x(2\nu_T u_x) + \partial_y(\nu_T u_y) + \partial_z(\nu_T u_z) + \partial_y(\nu_T v_x) + \partial_z(\nu_T w_x) \\
&= 2\nu_T u_{xx} + \nu_T u_{yy} + \nu_T u_{zz} + 2\partial_x\nu_T u_x + \partial_y\nu_T(u_y + v_x) + \partial_z\nu_T(u_z + w_x) + \nu_T(v_{xy} + w_{xz}) \\
&= \nu_T\Delta u + 2\partial_x\nu_T u_x + \partial_y\nu_T(u_y + v_x) + \partial_z\nu_T(u_z + w_x)
\end{aligned}$$

where we have used $v_{xy} + w_{xz} = -u_{xx}$ Thus we can write

$$\mathcal{D}_u = \nu_T\Delta u - 2\partial_x\nu_T(v_y + w_z) + \partial_y\nu_T(u_y + v_x) + \partial_z\nu_T(u_z + w_x)$$

This leads in two dimensions to the *curl-curl* form

$$\mathcal{D}_u^{(2D)} = \nu_T(-v_{xy} + u_{yy}) - 2\partial_x\nu_T v_y + \partial_y\nu_T(u_y + v_x) \tag{22}$$
$$\mathcal{D}_v^{(2D)} = \nu_T(\ \ v_{xx} - u_{xy}) - 2\partial_y\nu_T u_x + \partial_x\nu_T(v_x + u_y) \tag{23}$$

while in three dimensions,

$$\mathcal{D}_u = \nu_T(-v_{xy} - w_{xz} + u_{yy} + u_{zz}) - 2\partial_x\nu_T(v_y + w_z) + \partial_y\nu_T(u_y + v_x) + \partial_z\nu_T(u_z + w_x) \tag{24}$$

$$\mathcal{D}_v = \nu_T(\ v_{xx} - u_{xy} - w_{yz} + v_{zz}) - 2\partial_y\nu_T(w_z + u_x) + \partial_z\nu_T(v_z + w_y) + \partial_x\nu_T(v_x + u_y) \tag{25}$$

$$\mathcal{D}_w = \nu_T(\ w_{xx} + w_{yy} - u_{xz} - v_{yz}) - 2\partial_z\nu_T(u_x + v_y) + \partial_x\nu_T(w_x + u_z) + \partial_y\nu_T(w_y + v_z) \tag{26}$$

These *curl-curl* forms (22-26) remove the normal derivatives of the normal components of the velocity from $\mathbf{n} \cdot (\mathcal{D}_u, \mathcal{D}_v, \mathcal{D}_w)$. For example, the expression for $\mathcal{D}_u$ contains no x-derivatives of u while $\mathcal{D}_v$ contains no y-derivatives of v.

The alternative conservative form is

$$\begin{aligned}
\mathcal{D}_u &= \partial_x(-2\nu_T(v_y + w_z)) + \partial_y(\nu_T u_y) + \partial_z(\nu_T u_z) + \partial_y(\nu_T v_x) + \partial_z(\nu_T w_x) \\
\mathcal{D}_v &= \partial_x(\nu_T v_x) + \partial_y(-2\nu_T(u_x + w_z)) + \partial_z(\nu_T v_z) + \partial_x(\nu_T u_y) + \partial_z(\nu_T w_y) \\
\mathcal{D}_w &= \partial_x(\nu_T w_x) + \partial_y(\nu_T w_y) + \partial_z(-2\nu_T(u_x + v_y)) + \partial_y(\nu_T v_z) + \partial_x(\nu_T u_z)
\end{aligned}$$

9 Steady state line solver

We first consider the case of a rectangular grid in two space dimensions.

The implicit line solver uses local time stepping where the local time step $\Delta_{\mathbf{i}}$ is defined from

$$\Delta t_{\mathbf{i}} = ...$$

We solve implicit scalar-tri-diagonal systems in each spatial direction. Along the x-direction we solve a tridiagonal system for U, followed by a tridiagonal system for V for the equations

$$\begin{aligned}
\frac{U_{\mathbf{i}}^{n+1} - U_{\mathbf{i}}^{n}}{\Delta t_{\mathbf{i}}} &= -\Big\{ U^n D_{0x} U^{n+1} + V^n D_{0y} U^n + D_{0x} P^n \Big\} \\
&+ \nu \Big\{ D_{+x} D_{-x} U^{n+1} + (U_{j+1}^n - 2U^{n+1} + U_{j-1}^n)/h_y^2 \Big\} \\
&+ \nu_A(\mathbf{U}^n) \Big\{ \Delta_{+x} \Delta_{-x} U^{n+1} + (U_{j+1}^n - 2U^{n+1} + U_{j-1}^n) \Big\} \\
\frac{V_{\mathbf{i}}^{n+1} - V_{\mathbf{i}}^{n}}{\Delta t_{\mathbf{i}}} &= -\Big\{ U^n D_{0x} V^{n+1} + V^n D_{0y} V^n + D_{0y} P^n \Big\} \\
&+ \nu \Big\{ D_{+x} D_{-x} V^{n+1} + (V_{j+1}^n - 2V^{n+1} + V_{j-1}^n)/h_y^2 \Big\} \\
&+ \nu^{(2)}(\mathbf{U}^n) \Big\{ \Delta_{+x} \Delta_{-x} V^{n+1} + (V_{j+1}^n - 2V^{n+1} + V_{j-1}^n) \Big\}
\end{aligned}$$

Here $\nu^{(2)}(\mathbf{U}^n)$ is the coefficient of the artificial dissipation. There is also a self-adjoint version of the artificial dissipation,

$$\begin{aligned}
\beta_{SA} &= \Delta_{+x} \Big[\nu_{i_1-\frac{1}{2}}^{(2)} \Delta_{-x} \Big] U + \Delta_{+y} \Big[\nu_{i_2-\frac{1}{2}}^{(2)} \Delta_{-y} \Big] U \\
&= \nu_{i_1+\frac{1}{2}}^{(2)} (U_{i_1+1} - U) - \nu_{i_1-\frac{1}{2}}^{(2)} (U - U_{i_1-1}) \\
&+ \nu_{i_2+\frac{1}{2}}^{(2)} (U_{i_2+1} - U) - \nu_{i_2-\frac{1}{2}}^{(2)} (U - U_{i_2-1}) \\
&= \nu_{i_1+\frac{1}{2}}^{(2)} U_{i_1+1} + \nu_{i_1-\frac{1}{2}}^{(2)} U_{i_1-1} + \nu_{i_2+\frac{1}{2}}^{(2)} U_{i_2+1} + \nu_{i_2-\frac{1}{2}}^{(2)} U_{i_2-1} \\
&- \Big(\nu_{i_1+\frac{1}{2}}^{(2)} + \nu_{i_1-\frac{1}{2}}^{(2)} + \nu_{i_2+\frac{1}{2}}^{(2)} + \nu_{i_2-\frac{1}{2}}^{(2)} \Big) U
\end{aligned}$$

After solving in the x-direction we then solve along lines in the y-direction.

On curvilinear grids the expressions are a bit more complicated. Before discretization the equations transformed to the unit-square are

$$\begin{aligned}
u_t &= -\Big\{ (u r_x + v r_y) u_r + (u s_x + v s_y) u_s + r_x p_r + s_x p_s \Big\} \\
&+ \nu \frac{1}{J} \Big\{ \partial_r (J(r_x u_x + r_y u_y)) + \partial_s (J(s_x u_x + s_y u_y)) \Big\} \\
u_x &= r_x u_r + s_x u_s \\
u_y &= r_y u_r + s_y u_s \\
J &= |\partial \mathbf{x} / \partial \mathbf{r}| = x_r y_s - x_s y_r
\end{aligned}$$

We solve scalar-tridiagonal-systems in the r and s directions.

9.1 Fourth-order artificial dissipation

A fourth-order artificial dissipation is

$$\nu^{(4)}(\mathbf{U}^n)\Big\{(\Delta_{+x}\Delta_{-x})^2U + (\Delta_{+y}\Delta_{-y})^2U\Big\}$$

or in a self-adjoint form

$$\begin{aligned}
\beta_{SA}^{(4)} &= \Delta_{+x}\Delta_{-x}\Big[\nu_{\mathbf{i}}^{(4)}\Delta_{+x}\Delta_{-x}\Big]U + \Delta_{+y}\Delta_{-y}\Big[\nu_{\mathbf{i}}^{(4)}\Delta_{+y}\Delta_{-y}\Big]U \\
&= \nu_{i_1+1}^{(4)}\Delta_{+x}\Delta_{-x}U_{i_1+1} - 2\nu_{\mathbf{i}}^{(4)}\Delta_{+x}\Delta_{-x}U + \nu_{i_1-1}^{(4)}\Delta_{+x}\Delta_{-x}U_{i_1-1} \\
&+ \nu_{i_2+1}^{(4)}\Delta_{+y}\Delta_{-y}U_{i_2+1} - 2\nu_{\mathbf{i}}^{(4)}\Delta_{+y}\Delta_{-y}U + \nu_{i_2-1}^{(4)}\Delta_{+y}\Delta_{-y}U_{i_2-1} \\
&= \nu_{i_1+1}^{(4)}U_{i_1+2} - 2\big[\nu_{i_1+1}^{(4)} + \nu_{\mathbf{i}}^{(4)}\big]U_{i_1+1} + \nu_{i_1-1}^{(4)}U_{i_1-2} - 2\big[\nu_{i_1-1}^{(4)} + \nu_{\mathbf{i}}^{(4)}\big]U_{i_1-1} \\
&+ \nu_{i_2+1}^{(4)}U_{i_2+2} - 2\big[\nu_{i_2+1}^{(4)} + \nu_{\mathbf{i}}^{(4)}\big]U_{i_2+1} + \nu_{i_2-1}^{(4)}U_{i_2-2} - 2\big[\nu_{i_2-1}^{(4)} + \nu_{\mathbf{i}}^{(4)}\big]U_{i_2-1} \\
&+ \big[\nu_{i_1+1}^{(4)} + \nu_{i_2+1}^{(4)} + 8\nu_{\mathbf{i}}^{(4)} + \nu_{i_1-1}^{(4)} + \nu_{i_2-1}^{(4)}\big]U_{\mathbf{i}}
\end{aligned}$$

10 Convergence results

This section details the results of various convergence tests. Convergence results are run using the **twilight-zone** option, also known less formally as the **method of analytic solutions**. In this case the equations are forced so the the solution will be a known analytic function.

The tables show the maximum errors in the solution components. The rate shown is estimated convergence rate, σ, assuming $\text{error} \propto h^\sigma$. The rate is estimated by a least squares fit to the data.

The 2D trigonometric solution used as a twilight zone function is

$$
\begin{aligned}
u &= \frac{1}{2}\cos(\pi\omega_0 x)\cos(\pi\omega_1 y)\cos(\omega_3\pi t) + \frac{1}{2} \\
v &= \frac{1}{2}\sin(\pi\omega_0 x)\sin(\pi\omega_1 y)\cos(\omega_3\pi t) + \frac{1}{2} \\
p &= \cos(\pi\omega_0 x)\cos(\pi\omega_1 y)\cos(\omega_3\pi t) + \frac{1}{2}
\end{aligned}
$$

The 3D trigonometric solution is

$$
\begin{aligned}
u &= \cos(\pi\omega_0 x)\cos(\pi\omega_1 y)\cos(\pi\omega_2 z)\cos(\omega_3\pi t) \\
v &= \frac{1}{2}\sin(\pi\omega_0 x)\sin(\pi\omega_1 y)\cos(\pi\omega_2 z)\cos(\omega_3\pi t) \\
w &= \frac{1}{2}\sin(\pi\omega_0 x)\sin(\pi\omega_1 y)\sin(\pi\omega_2 z)\cos(\omega_3\pi t) \\
p &= \frac{1}{2}\sin(\pi\omega_0 x)\cos(\pi\omega_1 y)\cos(\pi\omega_2 z)\sin(\omega_3\pi t)
\end{aligned}
$$

When $\omega_0 = \omega_1 = \omega_2$ it follows that $\nabla\cdot\mathbf{u} = 0$. There are also algebraic polynomial solutions of different orders.

Tables (2-6) show results from running **OverBlown** on various grids.

grid	N	p	u	v	$\mathbf{u}$	$\nabla\cdot\mathbf{u}$
square20	20	2.9×10^{-1}	3.4×10^{-2}	3.4×10^{-2}	3.4×10^{-2}	5.0×10^{-1}
square30	30	9.6×10^{-2}	1.3×10^{-2}	1.2×10^{-2}	1.3×10^{-2}	1.8×10^{-1}
square40	40	4.5×10^{-2}	7.2×10^{-3}	6.7×10^{-3}	7.2×10^{-3}	8.1×10^{-2}
rate		2.69	2.23	2.34	2.24	2.61

Table 1: incompressible Navier Stokes, order=2, $\nu = 0.1$, $t = 1$, square, trig TZ, $\omega = 5.1$, $\alpha = 1$

grid	N	p	u	v	$\mathbf{u}$	$\nabla\cdot\mathbf{u}$
square16.order4	16	8.3×10^{-2}	1.2×10^{-2}	1.2×10^{-2}	1.2×10^{-2}	1.4×10^{-1}
square32.order4	32	6.5×10^{-3}	4.8×10^{-4}	3.9×10^{-4}	4.8×10^{-4}	7.6×10^{-3}
square64.order4	64	3.4×10^{-4}	2.6×10^{-5}	2.3×10^{-5}	2.6×10^{-5}	2.6×10^{-4}
rate		3.97	4.41	4.50	4.41	4.54

Table 2: incompressible Navier Stokes, order=4, $\nu = 0.1$, $t = 1$, square, trig TZ, $\omega = 5.1$, $\alpha = 1$

grid	N	p	u	v	$\nabla \cdot \mathbf{u}$
cic1	13	2.6×10^{-1}	1.5×10^{-1}	1.6×10^{-1}	5.2×10^{-1}
cic2	25	5.6×10^{-2}	2.4×10^{-2}	2.6×10^{-2}	1.3×10^{-1}
cic3	49	1.5×10^{-2}	5.1×10^{-3}	4.3×10^{-3}	2.3×10^{-2}
cic4	73	3.9×10^{-3}	1.3×10^{-3}	7.4×10^{-4}	4.9×10^{-3}
rate		2.4	2.7	3.0	2.7

Table 3: incompressible Navier Stokes, order=2, $\nu = 0.1$, $t = 1$, cic, trig TZ, $\omega = 2$

grid	N	p	u	v	$\mathbf{u}$	$\nabla \cdot \mathbf{u}$
cicb.order4	61	1.2×10^{-4}	3.8×10^{-5}	3.7×10^{-5}	3.8×10^{-5}	1.9×10^{-4}
cic.order4	121	9.6×10^{-6}	1.7×10^{-6}	2.1×10^{-6}	2.1×10^{-6}	9.6×10^{-6}
cic2.order4	241	6.2×10^{-7}	1.0×10^{-7}	1.2×10^{-7}	1.2×10^{-7}	1.0×10^{-6}
rate		3.86	4.30	4.18	4.19	3.78

Table 4: incompressible Navier Stokes, order=4, $\nu = 0.1$, $t = 1$, cic, trig TZ, $\omega = 1$, $\alpha = 1$

grid	N	p	u	v	w	$\nabla \cdot \mathbf{u}$
box10	10	2.0×10^{-2}	2.0×10^{-3}	2.1×10^{-3}	2.1×10^{-3}	1.7×10^{-2}
box20	20	5.0×10^{-3}	3.4×10^{-4}	3.1×10^{-4}	3.1×10^{-4}	2.7×10^{-3}
box30	30	2.4×10^{-3}	1.3×10^{-4}	1.0×10^{-4}	1.0×10^{-4}	8.1×10^{-4}
rate		1.9	2.5	2.8	2.8	2.8

Table 5: incompressible Navier Stokes, order=2, $\nu = 0.1$, $t = 1$, box, trig TZ, $\omega = 2$

grid	N	p	u	v	w	$\nabla \cdot \mathbf{u}$
box8.order4	8	1.4×10^{-3}	2.9×10^{-4}	2.6×10^{-4}	2.6×10^{-4}	1.2×10^{-3}
box16.order4	16	3.8×10^{-5}	8.4×10^{-6}	7.8×10^{-6}	7.8×10^{-6}	6.6×10^{-5}
box32.order4	32	3.6×10^{-6}	1.7×10^{-7}	1.8×10^{-7}	1.8×10^{-7}	1.6×10^{-6}
rate		4.3	5.4	5.2	5.2	4.8

Table 6: incompressible Navier Stokes, order=4, $\nu = 0.1$, $t = 1$, box, trig TZ, $\omega = 2$

grid	N	p	u	v	w	$\nabla \cdot \mathbf{u}$
sib1	17	2.8×10^{-1}	2.1×10^{-1}	1.4×10^{-1}	1.2×10^{-1}	6.2×10^{-1}
sib2	33	6.2×10^{-2}	5.0×10^{-2}	3.0×10^{-2}	1.9×10^{-2}	1.9×10^{-1}
sib2a	49	3.0×10^{-2}	1.7×10^{-2}	8.8×10^{-3}	1.2×10^{-2}	7.8×10^{-2}
rate		2.1	2.4	2.6	2.2	1.9

Table 7: incompressible Navier Stokes, order=2, $\nu = 0.1$, $t = 1$, sib, trig TZ, $\omega = 2$

grid	N	p	u	v	w	$\mathbf{u}$	$\nabla\cdot\mathbf{u}$
sib1.order4	30	1.0×10^{-2}	1.6×10^{-2}	8.8×10^{-3}	7.5×10^{-3}	1.6×10^{-2}	8.3×10^{-2}
sib1a.order4	60	8.2×10^{-4}	9.9×10^{-4}	4.5×10^{-4}	6.6×10^{-4}	9.9×10^{-4}	6.2×10^{-3}
sib2.order4	80	1.2×10^{-4}	8.7×10^{-5}	5.0×10^{-5}	7.8×10^{-5}	8.7×10^{-5}	7.8×10^{-4}
rate		4.38	5.08	5.10	4.44	5.08	4.57

Table 8: incompressible Navier Stokes, order=4, $\nu=0.05$, $t=1$, sib, trig TZ, $\omega=1$, $\alpha=1$

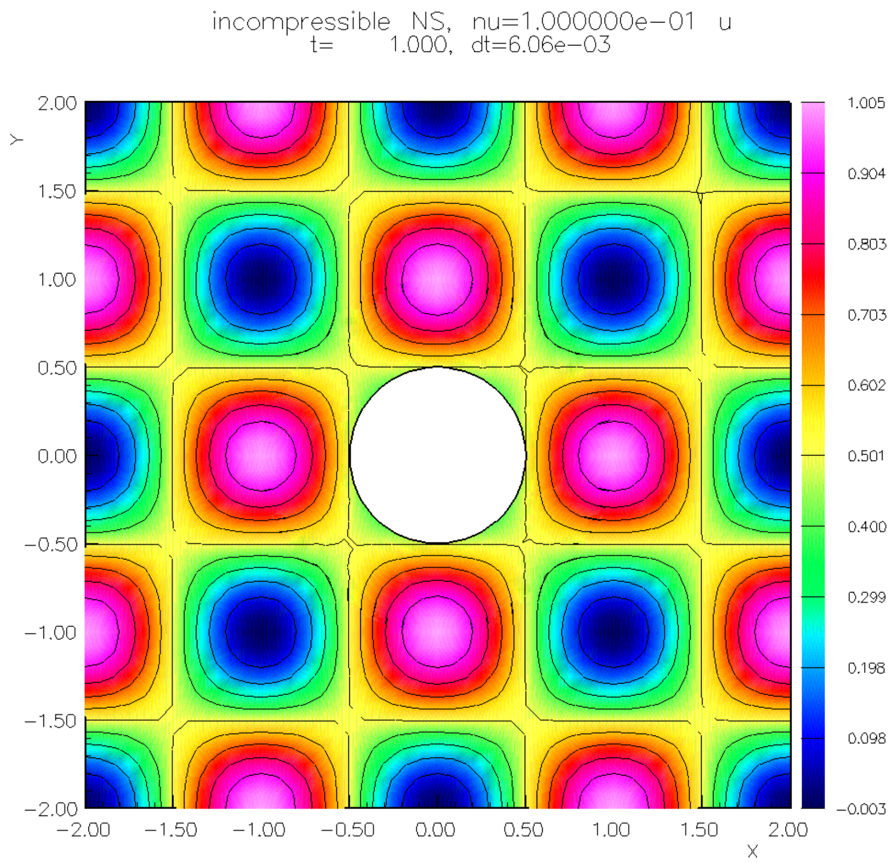


Figure 2: Incompressible N-S, twilight zone solution for convergence test

11 Some interesting examples

Here is a collection of interesting examples computed with the **OverBlown** incompressible solver.

11.1 Incompressible flow past a mast and sail

Figure (3) shows incompressible flow past a sail on a mast (grid created with `Overture/-sampleGrids/mastSail2d.cmd`.

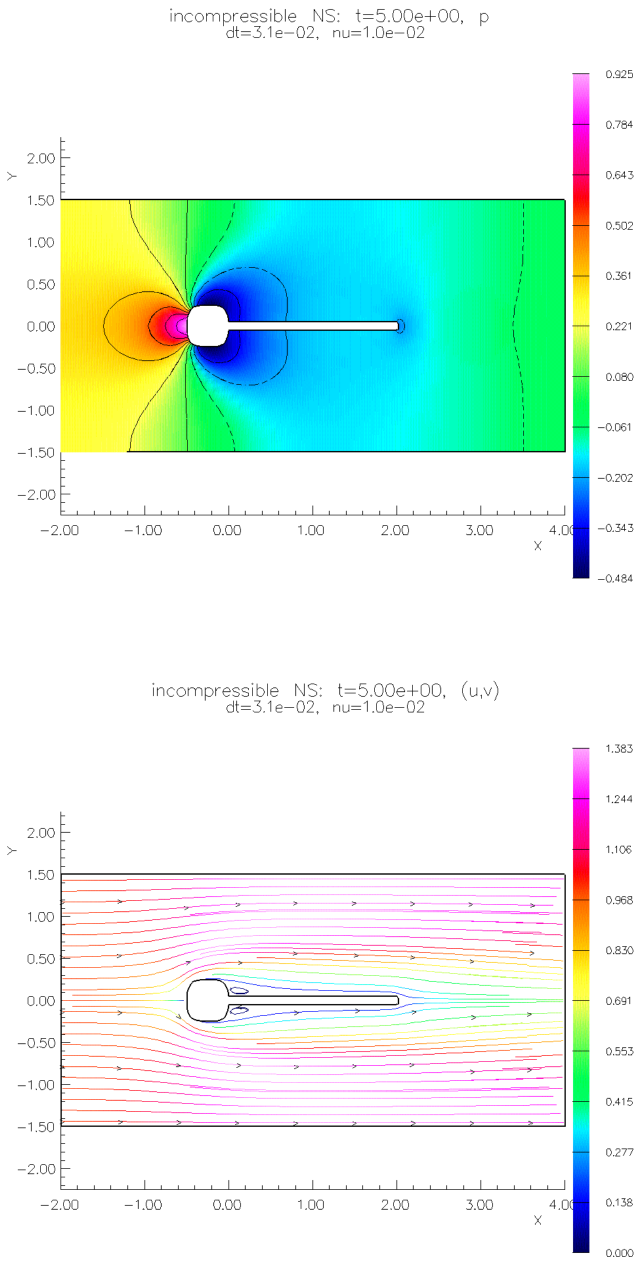


Figure 3: Incompressible flow past a mast and sail.

11.2 Two falling bodies in an incompressible flow

Figure (4) shows two rigid bodies failing under the influence of gravity in an incompressible flow.

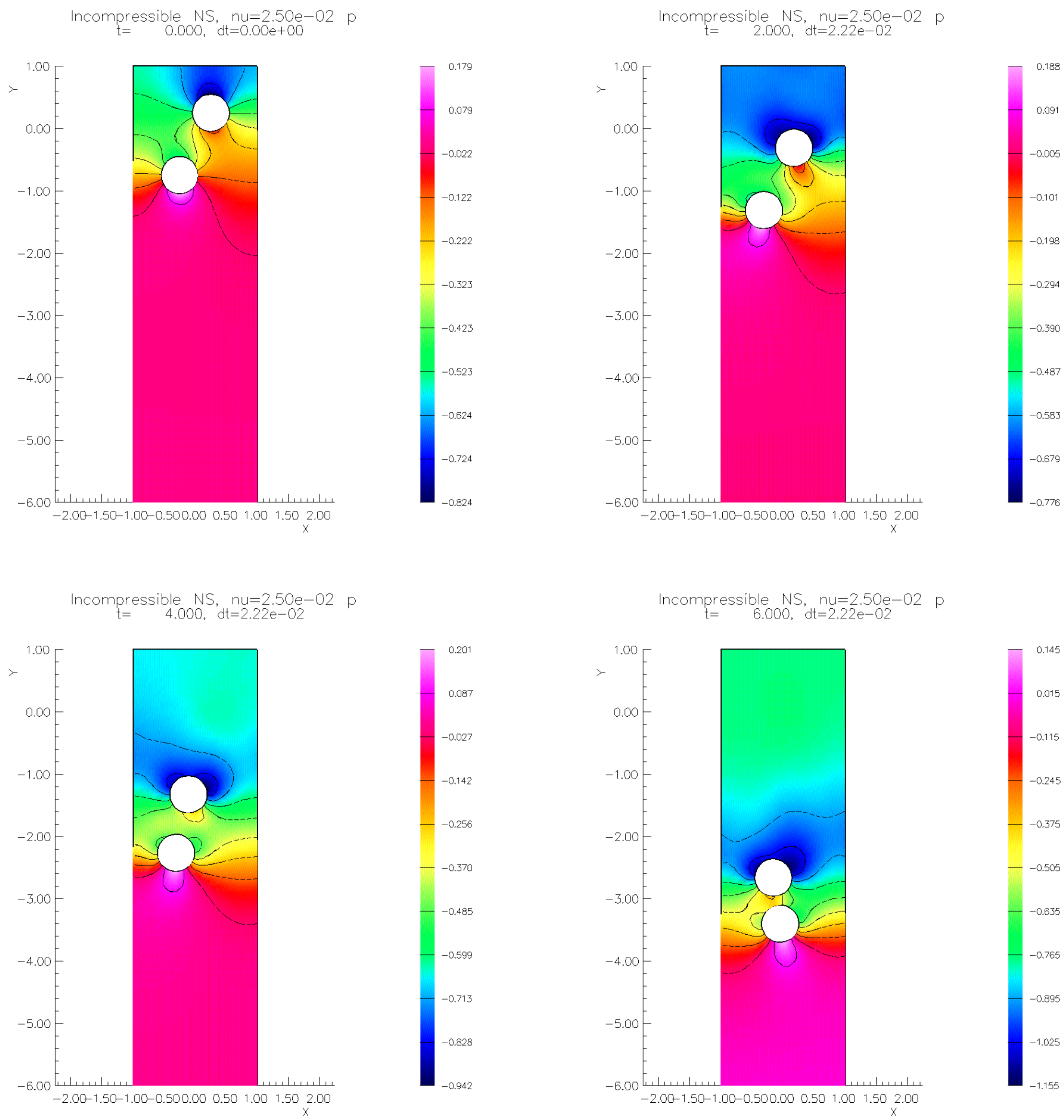


Figure 4: Two falling bodies in an incompressible flow.

11.3 Incompressible flow past a truck

Figure (5) shows a computation of the incompressible Navier-Stokes equations for flow past the cab of a truck. The steady state line solver was used for this computation.

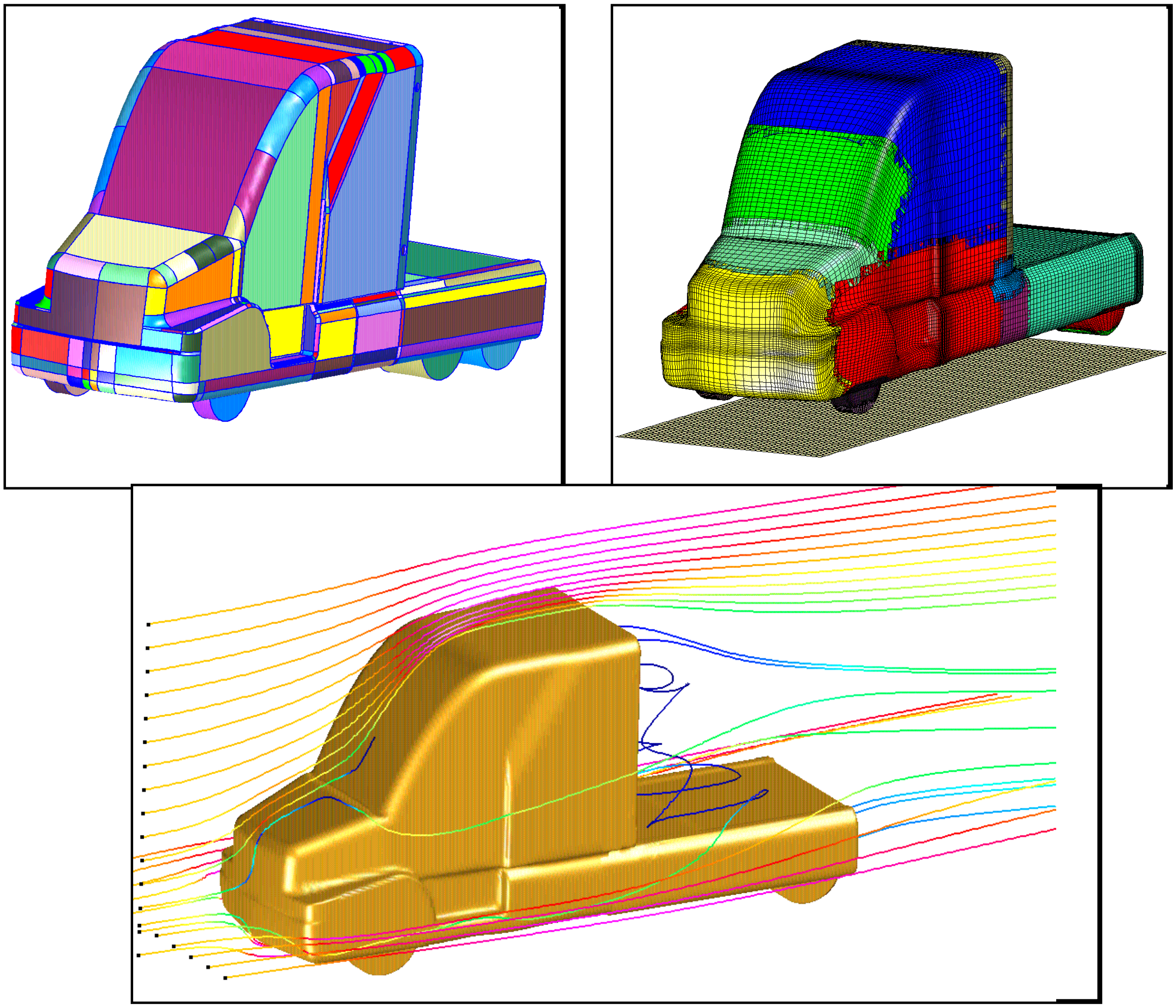
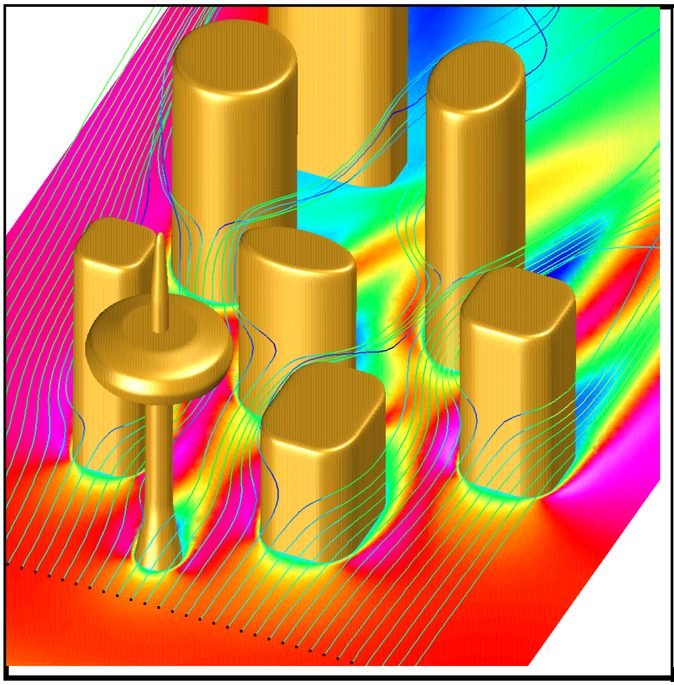


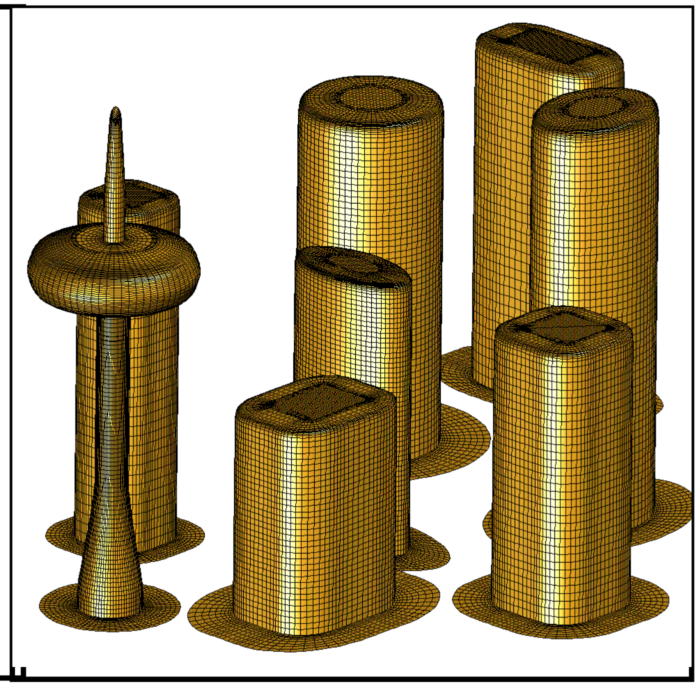
Figure 5: Incompressible flow past the cab of a truck. Shown are the CAD geometry, the grids and some tracer particles.

11.4 Incompressible flow past a city scape

Figure (11.4) shows a computation of the incompressible Navier-Stokes equations for flow past a city scape. The steady state line solver was used for this computation. The command file for generating this grid is `Overture/sampleGrids/multiBuildings.cmd` and the **OverBlown** command file is `OverBlown/ins/multiBuildings.cmd`.



3D flow past a city scape.



Overlapping grid for a city scape.

Notes:

◇ pseudo steady-state line implicit solver, 4th-order dissipation,
◇ local time-stepping (spatially varying dt)
◇ requires 1.4GB of memory,
◇ cpu = 59s/step,
◇ 2.2 GHz Xeon, 2 GB of memory

References

[1] D. L. BROWN, G. S. CHESSHIRE, W. D. HENSHAW, AND D. J. QUINLAN, *Overture: An object oriented software system for solving partial differential equations in serial and parallel environments*, in Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing, 1997.

[2] D. L. BROWN, W. D. HENSHAW, AND D. J. QUINLAN, *Overture: An object oriented framework for solving partial differential equations*, in Scientific Computing in Object-Oriented Parallel Environments, Springer Lecture Notes in Computer Science, **1343**, 1997, pp. 177–194.

[3] W. HENSHAW, *A fourth-order accurate method for the incompressible Navier-Stokes equations on overlapping grids*, J. Comp. Phys., 113 (1994), pp. 13–25.

[4] ——, *Overture: An object-oriented system for solving PDEs in moving geometries on overlapping grids*, in First AFOSR Conference on Dynamic Motion CFD, June 1996, L. Sakell and D. Knight, eds., 1996, pp. 281–290.

[5] W. HENSHAW AND H.-O. KREISS, *Analysis of a difference approximation for the incompressible Navier-Stokes equations*, Research Report LA-UR-95-3536, Los Alamos National Laboratory, 1995.

[6] W. HENSHAW, H.-O. KREISS, AND L. REYNA, *On the smallest scale for the incompressible Navier-Stokes equations*, Theoretical and Computational Fluid Dynamics, 1 (1989), pp. 65–95.

[7] ——, *Smallest scale estimates for the incompressible Navier-Stokes equations*, Arch. Rational Mech. Anal., 112 (1990), pp. 21–44.

[8] W. HENSHAW, H.-O. KREISS, AND L. REYNA, *A fourth-order accurate difference approximation for the incompressible Navier-Stokes equations*, Comput. Fluids, 23 (1994), pp. 575–593.

Index